KEY AGREEMENT AND TRANSPORT PROTOCOL

This application is a continuation-in-part of United States Application No. 98/426,090.

The present invention relates to key agreement protocols for transfer and authentication of encryption keys.

To retain privacy during the exchange of information it is well known to encrypt data using a key. The key must be chosen so that the correspondents are able to encrypt and decrypt messages but such that an interceptor cannot determine the contents of the message.

In a secret key cryptographic protocol, the correspondents share a common key that is secret to them. This requires the key to be agreed upon between the correspondents and for provision to be made to maintain the secrecy of the key and provide for change of the key should the underlying security be compromised.

Public key cryptographic protocols were first proposed in 1976 by Diffie-Hellman and utilized a public key made available to all potential correspondents and a private key known only to the intended recipient. The public and private keys are related such that a message encrypted with the public key of a recipient can be readily decrypted with the private key but the private key cannot be derived from the knowledge of the plaintext, ciphertext and public key.

Key establishment is the process by which two (or more) parties establish a shared secret key, called the session key. The session key is subsequently used to achieve some cryptographic goal, such as privacy. There are two kinds of key agreement protocol; key transport protocols in which a key is created by one party and securely transmitted to the second party; and key agreement protocols, in which both parties contribute information which jointly establish the shared secret key. The number of message exchanges required between the parties is called the number of passes. A key establishment protocol is said to provide implicit key authentication (or simply key authentication) if one party is assured that no other party aside from a specially identified second party may learn the value of the session key. The property of implicit key authentication does not necessarily mean that the second party actually

possesses the session key. A key establishment protocol is said to provide key confirmation if one party is assured that a specially identified second party actually has possession of a particular session key. If the authentication is provided to both parties involved in the protocol, then the key authentication is said to be mutual if provided to only one party, the authentication is said to be unilateral.

There are various prior proposals which claim to provide implicit key authentication.

Examples include the Nyberg-Rueppel one-pass protocol and the Matsumoto-Takashima-Imai (MTI) and the Goss and Yacobi two-pass protocols for key agreement.

The prior proposals ensure that transmissions between correspondents to establish a common key are secure and that an interloper cannot retrieve the session key and decrypt the ciphertext. In this way security for sensitive transactions such as transfer of funds is provided.

For example, the MTI/A0 key agreement protocol establishes a shared secret K, known to the two correspondents, in the following manner:-

1. During initial, one-time setup, key generation and publication is undertaken by selecting and publishing an appropriate system prime p and generator in a manner guaranteeing authenticity. Correspondent A selects as a long-term private key a random integer "a",$1 \leq a \leq p-2$, and computes a long-term public key $z_A = \alpha^a \bmod p$. B generates analogous keys b, z_B. A and B have access to authenticated copies of each other's long-term public key.

2. The protocol requires the exchange of the following messages.

$$A \rightarrow B: \alpha^x \bmod p \ (1)$$

$$A \leftarrow B: \alpha^y \bmod p \ (2)$$

The values of x and y remain secure during such transmissions as it is impractical to determine the exponent even when the value of α and the exponentiation is known provided of course that p is chosen sufficiently large.

3. To implement the protocol the following steps are performed each time a shared key is required.

(a) A chooses a random integer $x, 1 \leq x \leq p-2$, and sends B message (1) i.e. $\alpha^x \bmod p$.

(b) B chooses a random integer $y, 1 \leq y \leq p-2$, and sends A message (2) i.e. $\alpha^y \bmod p$.

(c) A computes the key $K = (\alpha^y)^a z_B{}^x \bmod p$.

(d) B computes the key $K = (\alpha^x)^b z_A{}^y \bmod p$.

(e) Both share the key $K - \alpha^{bx+ay}$.

In order to compute the key K, A must use his secret key a and the random integer x, both of which are known only to him. Similarly B must use her secret key b and random integer y to compute the session key K. Provided the secret keys a,b remain uncompromised, an interloper cannot generate a session key identical to the other correspondent. Accordingly, any ciphertext will not be decipherable by both correspondents.

As such this and related protocols have been considered satisfactory for key establishment and resistant to conventional eavesdropping or man-in-the-middle attacks.

In some circumstances it may be advantageous for an adversary to mislead one correspondent as to the true identity of the other correspondent.

In such an attack an active adversary or interloper E modifies messages exchanged between A and B, with the result that B believes that he shares a key K with E while A believes that she shares the same key K with B. Even though E does not learn the value of K the misinformation as to the identity of the correspondents may be useful.

A practical scenario where such an attack may be launched successfully is the following. Suppose that B is a bank branch and A is an account holder. Certificates are issued by the bank headquarters and within the certificate is the account information of the holder. Suppose that the protocol for electronic deposit of funds is to exchange a key with a bank branch via a mutually authenticated key agreement. Once B has authenticated the transmitting entity, encrypted funds are

deposited to the account number in the certificate. If no further authentication is done in the encrypted deposit message (which might be the case to save bandwidth) then the deposit will be made to E's account.

It is therefore an object of the present invention to provide a protocol in which the above disadvantages are obviated or mitigated.

According therefore to the present invention there is provided a method of authenticating a pair of correspondents A,B to permit exchange of information therebetween, each of said correspondents having a respective private key a,b and a public key p_A, p_B derived from a generator α and respective ones of said private keys a,b, said method including the steps of

i) a first of said correspondents A selecting a first random integer x and exponentiating a function $f(\alpha)$ including said generator to a power $g^{(x)}$ to provide a first exponentiated function $f(\alpha)^{g(x)}$;

ii) said first correspondent A forwarding to a second correspondent B a message including said first exponentiated function $f(\alpha)^{g(x)}$;

iii) said correspondent B selecting a second random integer y and exponentiating a function $f'(\alpha)$ including said generator to a power $g^{(y)}$ to provide a second exponentiated function $f'(\alpha)^{g(y)}$;

iv) said second correspondent B constructing a session key K from information made public by said first correspondent A and information that is private to said second correspondent B, said session key also being constructible by said first correspondent A for information made public by B and information that is private to said first correspondent A;

v) said second correspondent B generating a value h of a function $F[\delta,K]$ where $F[\delta,K]$ denotes a cryptographic function applied conjointly to δ and K and where δ is a subset of the public information provided by B thereby to bind the values of δ and K;

vi) said second of said correspondents B forwarding a message to said first correspondent A including said second exponential function $f'(\alpha)^{g(y)}$ and said value h of said cryptographic function $F[\delta,K]$;

vii) said first correspondent receiving said message and computing a session key K' from information made public by said second correspondent B and private to said first correspondent A;

viii) said first correspondent A computing a value h' of a cryptographic function h,h' F[δ,K']; and

ix) comparing said values obtained from said cryptographic functions F to confirm their correspondence.

As the session key K can only be generated using information that is private to either A or B, the binding of K with δ with the cryptographic function h prevents E from extracting K or interjecting a new value function that will correspond to that obtained by A.

Embodiments of the invention will now be described by way of example only with reference to the accompanying drawings in which.

Figure 1 is a schematic representation of a data communication system.

Figures 2 through 7 are schematic representations of implementations of different protocols.

Referring therefore to Figure 1, a pair of correspondents, 10,12, denoted as correspondent A and correspondent B, exchange information over a communication channel 14. A cryptographic unit 16,18 is interposed between each of the correspondents 10,12 and the channel 14. A key 20 is associated with each of the cryptographic units 16,18 to convert plaintext carried between each unit 16,18 and its respective correspondent 10,12 into ciphertext carried on the channel 14.

In operation, a message generated by correspondent A, 10, is encrypted by the unit 16 with the key 20 and transmitted as ciphertext over channel 14 to the unit 18.

The key 20 operates upon the ciphertext in the unit 18 to generate a plaintext message for the correspondent B, 12. Provided the keys 20 correspond, the message received by the correspondent 12 will be that sent by the correspondent 10.

In order for the system shown in Figure 1 to operate it is necessary for the keys 20 to be identical and therefore a key agreement protocol is established that

allows the transfer of information in a public manner to establish the identical keys. A number of protocols are available for such key generation and embodiments of the present invention will be described below in the context of modifications of existing protocols.

A commonly used set of protocols are collectively known as the Matsumoto-Takashima-Imai or "MTI" key agreement protocols, and are variants of the Diffie-Hellman key exchange. Their purpose is for parties A and B to establish a secret session key K.

The system parameters for these protocols are a prime number p and a generator α of the multiplicative group

. Correspondent A has private key a and public key $p_A = \alpha^a$. Correspondent B has private key b and public key $p_B = \alpha^b$. In all four protocols exemplified below, $text_A$ refers to a string of information that identifies party A. If the other correspondent B possesses an authentic copy of correspondent A's public key, then $text_A$ will contain A's public-key certificate, issued by a trusted center; correspondent B can use his authentic copy of the trusted center's public key to verify correspondent A's certificate, hence obtaining an authentic copy of correspondent A's public key.

In each example below it is assumed that an interloper E wishes to have messages from A identified as having originated from E herself. To accomplish this, E selects a random integer e, $1 \leq e \leq p-2$, computes $p_E=(p_A)^e=\alpha^{ae}$ mod p, and gets this certified as her public key. E does not know the exponent ae, although she knows e. By substituting $text_E$ for $text_A$, the correspondent B will assume that the message originates from E rather than A and use E's public key to generate the session key K. E also intercepts the message from B and uses his secret random integer e to modify its contents. A will then use that information to generate the same session key allowing A to communicate with B.

The present invention is exemplified by modifications to 4 of the family of MTI protocols which foil this new attack thereby achieving the desired property of mutual implicit authentication. In the modified protocols exemplified below F(X,Y) denotes a cryptographic function applied to a string derived from x and

y. Typically and as exemplified a hash function, such as the NIST "Secure Hash Algorithm"(SHA-1), is applied to the string obtained by concatenating X and Y but it will be understood that other cryptographic functions may be used.

Example 1 - MTI/A0 protocol

The existing protocol operates as follows:-

1. Correspondent A generates a random integer x, $1 \le x \le p-2$, computes α^x, and sends $\{\alpha^x, text_A\}$ to party B.
2. Correspondent B generates a random integer y, $1 \le y \le p-2$, computes α^y, and sends $\{\alpha^y, text_B\}$ to party A.
3. Correspondent A computes $K = (\alpha^y)^a (p_B)^x = \alpha^{ay+bx}$.
4. Correspondent B computes $K = (a^x)^b (p_A)^y = \alpha^{ay+bx}$.

A common key K is thus obtained. However, with this arrangement, interloper E may have messages generated by correspondent A identified as having originated from E in the following manner.

1. E intercepts A's message $\{\alpha^x, text_A\}$ and replaces it with $\{\alpha^x, text_E\}$.

The provision of the message $text_E$ identifies the message as having originated at E.

2. B sends $\{\alpha^y, text_B\}$ to E, who then forwards $\{(\alpha^y)^e, text_B\}$ to A. Since A receives $text_B$, he assumes the message originates at B and, as he does not know the value of y, assumes that α^{ye} is valid information.
3. A computes $K = (\alpha^{cy})^a (p_B)^x = \alpha^{acy+bx}$.
4. B computes $K = (\alpha^x)^b (p_E)^y = \alpha^{acy+bx}$.
5. A and B now share the key K, even though B believes he shares a key with E.

Accordingly any further transactions from A to B will be considered by B to have originated at E. B will act accordingly crediting instruction to E. Even though the interloper E does not learn the value of the session key K nevertheless the assumption that the message originates at E may be valuable and achieve the desired effect.

To avoid this problem, the protocol is modified as follows:-

1. A generates a random integer $x, 1 \leq x \leq p-2$, computes α^x, and sends $\{\alpha^x, text_A\}$ to party B.

2. B generates a random integer $y, 1 \leq y \leq p-2$, and computes α^y, $K = (\alpha^x)^b (p_A)^y = \alpha^{ay+bx}$, and a value h of cryptographic hash function $F(\alpha^y, \alpha^{ay+bx})$ which is a function of public information δ and the key K. B sends $\{\alpha^y, h, text_B\}$ to party A.

3. A computes $K = (\alpha^y)^a (p_B)^x = \alpha^{ay+bx}$. A also computes a value h' of cryptographic hash function $F(\alpha^y, K)$ and verifies that this value is equal to h.

If E attempts to interpose her identification, $text_E$, the attack fails on the modified protocols because in each case B sends the hash value $F(\delta, K)$, where δ is B's random exponential, α^y, thereby binding together the values of δ and K. E cannot subsequently replace the value of δ with δ^e and compute $F(\delta^e, K)$ since E does not know K. Even though E knows α^y, this is not sufficient to extract K from the hash value h. Accordingly, even if E interposes the value α^{ye} so that the keys 20 will agree, the values h,h' will not.

Example 2 - MTI/B0 protocol

In this protocol,

1. A generates a random integer $x, 1 \leq x \leq p-2$, computes $(p_B)^x = \alpha^{bx}$, and sends $\{\alpha^{bx}, text_A\}$ to party B.

2. B generates a random integer $y, 1 \leq y \leq p-2$, computes $(p_A)^y = \alpha^{ay}$, and sends $\{\alpha^{ay}, text_B\}$ to party A.

3. A computes $K = (\alpha^{ay})^{a^{-1}} \alpha^x = \alpha^{x+y}$

4. B computes $K = (\alpha^{bx})^{b^{-1}} \alpha^y = \alpha^{x+y}$

This protocol is vulnerable to the interloper E if,

1. E replaces A's message $\{\alpha^{bx}, text_A\}$ with $\{\alpha^{bx}, text_E\}$ to identify herself as the originator to the message.

2. B sends $\{(p_E)^y, text_B\}$ to E, who then computes

$((P_E)^y)^{e^{-1}} = \alpha^{ay}$ and forwards $\{\alpha^{ay}, text_B\}$ to A.

3. A computes $K = (\alpha^{ay})^{a^{-1}}\alpha^x = \alpha^{x+y}$

4. B computes $K = (\alpha^{bx})^{b^{-1}}\alpha^y = \alpha^{x+y}$

5. A and B now share the key K, even though B believes he shares a key with E.

This protocol may be modified to resist E's attack as follows.

1. A generates a random integer $x, 1 \le x \le p-2$, computes $(p_B)^x = \alpha^{bx}$, and sends $\{\alpha^{bx}, text_A\}$ to party B.
2. B generates a random integer y, $1 \le x \le p-2$, and computes $(p_A)^y = \alpha^{ay}$, $K=(\alpha^{bx})\ \alpha^y = \alpha^{x+y}$, and the value h of hash function $F(\alpha^y = \alpha^{x+y})$. B sends $\{\alpha^{ay}, h, text_B\}$ to A.
3. A computes $K=(\alpha^{ay})\ \alpha^x = \alpha^{x+y}$. A also computes the value h' of hash function F (α^{ay}, K) and verifies that this value is equal to h.

Once again, E cannot determine the session key K and so cannot generate a new value of the hash function to maintain the deception.

Example 3 - MTI/CO protocol

This protocol operates as follows:-

1. A generates a random integer $x, 1 \le x \le p-2$, computes $(p_B)^x = \alpha^{bx}$, and sends $\{\alpha^{bx}, text_A\}$ to party B.
2. B generates a random integer $y, 1 \le y \le p-2$, computes $(p_A)^y = \alpha^{ay}$, and sends $\{\alpha^{ay}, text_B\}$ to party A.
3. A computes $K = (\alpha^{ay})^{a^{-1}x} = \alpha^{xy}$
4. B computes $K = (\alpha^{bx})^{b^{-1}y} = \alpha^{xy}$

The interloper E may interpose her identity as follows:-

1. E replaces A's message $\{\alpha^{bx}, text_A\}$ with $\{\alpha^{bx}, text_E\}$.
2. B sends $\{(p_E)^y, text_B\}$ to E, who then computes $((p_E)^y)^{e-1} = \alpha^{ay}$ and forwards $\{\alpha^{ay}, text_B\}$ to A.

3. A computes $K = (\alpha^{ay})^{a^{-1}x} = \alpha^{xy}$

4. B computes $K = (\alpha^{bx})^{b^{-1}y} = \alpha^{xy}$

5. A and B now share the key K, even though B believes he shares a key with E.

To avoid this attack protocol is modified as follows:-

1. A generates a random integer $x, 1 \leq x \leq p-2$, computes $(p_B)^x = \alpha^{bx}$, and sends $\{\alpha^{bx}, text_A\}$ to party B.

2. B generates a random integer $y, 1 \leq y \leq p-2$, and computes $(p_A)^y = \alpha^{ay}$, $K = (\alpha^{bx})^{b^{-1}y} = \alpha^{xy}$, and value h of hash function $F(\alpha^{ay}, \alpha^{xy})$. B sends $\{\alpha^{ay}, h, text_B\}$ to party A.

3. A computes $K = (\alpha^{ay})^{a^{-1}x} = \alpha^{xy}$. A also computes the value h' of $F(\alpha^{ay}, K)$ and verifies that this value is equal to h.

Example 4 - MTI/C1 protocol

In this protocol:-

1. A generates a random integer $x, 1 \leq x \leq p-2$, computes $(p_B)^{ax} = \alpha^{abx}$, and sends $\{\alpha^{abx}, text_A\}$ to party B.
2. B generates a random integer $y, 1 \leq y \leq p-2$, computes $(p_A)^{by} = \alpha^{aby}$, and sends $\{\alpha^{aby}, text_B\}$ to party A.
3. A computes $K = (\alpha^{aby})^x = \alpha^{abxy}$.
4. B computes $K = (\alpha^{abx})^y = \alpha^{abxy}$.

E can act as an interloper as follows:-

1. E replaces A's message $\{\alpha^{abx}, text_A\}$ with $\{\alpha^{abx}, text_E\}$.
2. B sends $\{(p_E)^{by}, text_B\}$ to E, who then computes $((p_E)^{by})^{c^{-1}} = \alpha^{aby}$ and forwards $\{\alpha^{aby}, text_B\}$ to A.
3. A computes $K = (\alpha^{aby})^x = \alpha^{abxy}$.
4. B computes $K = (\alpha^{abx})^y = \alpha^{abxy}$.
5. A and B now share the key K, even though B believes he shares a key with E.

To avoid this, the protocol is modified as follows:-

1. A generates a random integer $x, 1 \leq x \leq p-2$, computes $(p_B)^{ax} = \alpha^{abx}$, and sends $\{\alpha^{abx}, text_A\}$ to party B.
2. B generates a random integer $y, 1 \leq y \leq p-2$, and computes $(p_A)^{by} = \alpha^{aby}$, $K = (\alpha^{abx})^y = \alpha^{abxy}$, and $h = F(\alpha^{aby}, \alpha^{abxy})$. B sends $\{\alpha^{aby}, h, text_B\}$ to party A.
3. A computes $K = (\alpha^{aby})^x = \alpha^{abxy}$. A also computes $h' = F(\alpha^{aby}, K)$ and verifies that this value is equal to h.

In each of the modified protocols discussed above, key confirmation from B to A is provided.

As noted above instead of F being a cryptographic hash function other

functions could be used. For example, an option available is to choose $F = \in_K$, where $\in$ is the encryption function of a suitable symmetric-key encryption scheme, and K is the session key established. Because E cannot generate the session key K, it is similarly not able to generate the value of the function F and therefore cannot interpose for the correspondent A.

The technique described above can be applied to other similar key exchange protocols, including all of the 3 infinite classes of MTI protocols called MTI-A(k), MTI-B(k) and MTI-C(k).

The Goss authenticated key exchange protocol is similar to the MTI/A0 protocol, except that the session key is the bitwise exclusive-OR of α^{ay} and α^{bx}; that is $K = \alpha^{ay} \oplus \alpha^{bx}$ instead of being the product of α^{ay} and α^{bx}. Hence the attack on the MTI/A0 protocol and its modification can be extended in a straightforward manner to the case of the Goss protocol.

Similarly Yacobi's authenticated key exchange protocol is exactly the same as the MTI/A0 protocol, except that α is an element of the group of units , where n is the product of 2 large primes. Again, the attack on the MTI/A0 protocol and its modification can be extended in a straightforward manner to the case of the Goss protocol.

A further way of foiling the interposition of E is to require that each entity prove to a trusted center that it knows the exponent of α that produces its public key P, before the center issues a certificate for the public key. Because E only knows "e" and not "ae" it would not meet this requirement. This can be achieved through zero knowledge techniques to protect the secrecy of the private keys but also requires the availability of a trusted centre which may not be convenient.

Each of the above examples has been described with a 2 pass protocol for key authentication. One pass protocols also exist to establish a key between correspondents and may be similarly vulnerable.

As an example the Nyberg-Rueppel one pass key agreement protocol will be described and a modification proposed.

The purpose of this protocol is for party A and party B to agree upon a

secret session key K.

The system parameters for these protocols are a prime number p and a generator α of the multiplicative group $\alpha \varepsilon Z_p^*$. User A has private key a and public key $p_A = \alpha^a$. User B has private key b and public key $p_B = \alpha^b$.

1. A selects random integers x and t, $1 \leq x,t \leq p-2$.
2. B recovers the value α^x mod p by computing $\alpha^s (p_A)^r$ mod p and then computes the shared session key $K=(r\,\alpha x)^{b^{-1}} = \alpha^t$ mod p.

If interloper E wishes to have messages from A identified as having originated from herself, E selects a random integer e, $1 \leq e \leq p-2$, computes $p_E = \alpha^e$, and gets this certified as her public key.

1. E intercepts A's message $\{r,s,text_A\}$ and computes $\alpha^x = \alpha^s(p_A)^r$ and $\alpha^{bt} = r\alpha^x$.
2. E then selects a random integer x', $1 \leq x' \leq p-2$, computes $r'=\alpha bt\, \alpha^{-x'}$ mod p and $s'=x'-r'e$ mod (p-1).
3. E sends $\{r',s',text_E\}$ to B.
4. B recovers the value mod p by computing $\alpha^{s'}(p_E)^{r'}$ mod p and then computes $K = (r'\alpha^{x'})^{b^{-1}} = \alpha^t$ mod p.
5. A and B now share the key K, even though B believes he shares a key with E.

To foil such an attack the protocol is modified by requiring A to also transmit a value h of $F(p_A,K)$, where F is a hash function, an encryption function of a symmetric-key system with key K or other suitable cryptographic function. The modified protocol is the following.

1. A selects random integers x and t, $1 \leq x,t \leq p-2$.
2. A computes $r = (p_B)^t\alpha^{-x}$ mod p, $s = x - ra$ mod (p-1), session key $K = \alpha^t$ mod p and the value h of hash function $F(p_A,K)$. A sends $\{r,s,h,text_A\}$ to B.

3. B recovers the value α^x mod p by computing $\alpha^s(p_A)^r$ mod p and then computes the shared session key $K=(r\alpha^x)^{b^{-1}} = \alpha^t$ mod p. B also computes the value h' of function $F(p_A,K)$ and verifies that this value is equal to h.

Again therefore by binding together the public information π and the session key K in the hash function, the interposition of E will not result in identical hash functions h,h'.

In each case it can be seen that a relatively simple modification to the protocols involving the binding of public and private information in a cryptographic function foils the interposition of interloper E.

All the protocols discussed above have been described in the setting of the multiplicative group $\alpha\varepsilon Z_p^*$. However, they can all be easily modified to work in any finite group in which the discrete logarithm problem appears intractable. Suitable choices include the multiplicative group of a finite field (in particular the finite field $GF(2^n)$), subgroups of $\alpha\varepsilon Z_p^*$ of order q, and the group of points on an elliptic curve defined over a finite field. In each case an appropriate generator α will be used to define the public keys.

The protocols discussed above can also be modified in a straightforward way to handle the situation when each user picks their own system parameters p and α (or analogous parameters if a group other than Z_p^* is used).

Further implementations are shown schematically in figures 2 through 7. A different notation is utilized but it will be understood that this notation may be mapped to that of the previous embodiments.

Referring to figure 2, a mutual public key authenticated key agreement protocol is complemented between a correspondent A shown on the left hand side of the figure and a correspondent B shown on the right hand side. Correspondent A has a public-private key pair P_A,S_A respectively and similarly correspondent B has a public private

Key pair P_B,S_B.

As a first step, correspondent A generates a session private key as a random number RND_A and computes a corresponding public session key $G_A = F_A(RND_A)$. The function F_A is a cryptographic one way function, typically an exponention by the group generator, such as a point multiplication in an elliptic curve cryptosystem.

The public session key G_A is forwarded to correspondent B who generates corresponding parameters of a session private key RND_B and the exponent G_B.

The correspondent B computes a session key K as a function of A's public information G_A,P_A AND B's private information RND_B,S_B. A corresponding key K' can be computed by A using the private information of A and the public information of B namely $f(RND_A,G_B,S_A,P_B)$.

After correspondent B has generated the key K, he compiles a string $(G_A//G_B//Id_A)$ where Id_A is a string that identifies A. The concatenated string is hashed with a cryptographic function h_k which is a keyed hash function that uses the key K to yield a string $hash_B$.

The string $hash_B$ is forwarded to correspondent A together with Id_A and G_B.

Upon receipt of the message from B, correspondent A computes the key K' as described above. Correspondent A also computes a hash, $hashverify_B$ from the string $(G_B//G_A//Id_A)$ using the hash function keyed by the key K'. correspondent A checks that the hashes verify to confirm the identity of the keys K,K'.

Correspondent A then computes a hash h_K using the key K on the string $(G_A//G_B//Id_B)$ and forwards that together with Id_B correspondent B. Correspondent B similarly computes a $hashverify_A$ using the keyed hash function h_K on the same string and

verifies that $hash_A = hashverify_A$.

A similar protocol is shown in figure 3 to implement a mutual symmetric key authentication protocol. In this protocol the correspondents share a key K obtained over a secure channel. The correspondents A.B, each generate a random integer which is used as the session public key of A and B respectively. Thereafter the exchange of information and verification proceeds as above with respect to figure 2 with the shared secret key being utilised in the keyed hash functions.

A full mutual public key authenticated protocol is shown in figure 4. An initial exchange of the public keys P_A,P_B is performed over an authenticated channel followed by the exchange of information as shown in the protocol of figure 4. In this case the correspondent A sends G_A computed as described above with respect to figure 2, together with a string x_2 that A wants confirmation of receipt by B. Correspondent B computes the key K as in figure 2 and also generates a pair of strings y_1,y_2 which B wants to have authenticated by A and receipt confirmed by A respectively. The strings are sent to A with the hash $hash_B$ and identity Id_A..The hash $hash_B$ is performed on a string including the message x_2 and the string y_1 wants authenticated.

Correspondent A computes the key K and verifies the hash as before. This also confirms receipt of x_2 by B.

Correspondent A in turn generates strings z_1,z_2 where z_1 is a string that A wants authenticated by B and z_2 is a string that may be used in a subsequent stage of the protocol described below. The strings, z_1 and y_2 together with the identifying information of B, Id_B, are included in the string that is hashed with the key K to provide the string $hash_A$. this is sent together with the identitiy of B and the strings z_1,z_2 to the correspondent B who can verify the hashes as before, thereby confirming receipt of y_2 and authenticating z_1.

Thus the exchange of information is exchanged in an authenticated manner and a common key obtained that allows subsequent exchange of correspondence on a secure channel.

With the protocol described in figure 4 it is possible to implement a mutual public key authenticated key agreement protocol by letting the strings x_2, y_1, y_2, z_1, z_2 all be empty strings. Alternatively, a mutual public key authenticated key agreement protocol with key transport can be implemented by using x_2 as a string that is assumed to represent $E_K(k)$. Correspondent B can compute the value of K and hence retrieve the notional value of k from the string. He can use this as his CRP,. The values of y_1 may be used to represent $E_K(k_{21})$ and z_1 as $E_K(k_{12})$ where k_{21} and k_{12} are different keys for communication or other secret information to be shared between the correspondents. In this case y_1 and z_2 are empty strings. In this way there is a key agreement on a shared key K_{AB} together with authenticated key transport of the keys k_{21} and k_{12}between the correspondents. Moreover, if additional information is provided in the x_2 and y_2 then confirmation of proper receipt is also obtained.

The protocol of figure 4 may also be used to increase efficiency in successive sessions by using the string z_2 to pass the information exchanged in the first pass of the next session. Thus as shown in figure 5, the string G_A, x_2 is sent as z_2 in the previous session. The protocol then proceeds from correspondent B as before. Correspondent B may also take advantage of this facility by including the information G_B, y_1 for the next session in the exchange as y_2.

The mutual public key authenticated key agreement protocol may also be adapted for symmetric key implementations as shown in figure 6. In this case, as in figure 3 above, the key generation is omitted as the correspondents have a shared key obtained over a secure channel.

Similarly, the protocol of figure 6 may be modified as illustrated in figure 7 to take advantage of the exchange of information in a previous session, similar to that of figure 5.

It will be seen therefore that a number of versatile and flexible protocols can be developed from the general protocol to meet particular needs. These protocols may implement elliptic curve cryptography or operate in Z_p as preferred.